Fujitsu Semiconductor (Shanghai) Co., Ltd.
Application Note

MCU-AN- 500066-E-11

F²MC-8FX FAMILY

8-BIT MICROCONTROLLER

MB95310/370 SERIES

INFRARED REMOTE FUNCTION API

APPLICATION NOTE

FUJITSU

Revision History

Date	Author	Change of Records
2009-11-06	Jane Li	V1.0, First draft
2011-03-17	Jane Li	V1.1, modified Chinese remark

This manual contains 20 pages.

CONTENTS

1 Introduction

This document introduces API for infrared remote function.

Infrared remote function is generally used in TV, audio system and air conditioner. In following chapters we describe theory and library of infrared remote. Chapter three is the theory of infrared remote and chapter four is function library of infrared remote. In function library we should set up three functions to control infrared remote, Initials remote interrupt, remote interrupt function to receive infrared code and key judge function to distinguish which key pressed.

2 Background

This chapter introduces background of infrared remote.

Infrared remote is widely used in TV, VCD, DVD, air condition and so on， because it is easy to be used, easy to be bought and it is very cheap.

Infrared remote has occurred about twenty five years. It encodes the pressed key and transfers this key value to “0” and “1” and then sends out; because each key has the only code so user can judge the key value according to the code. So user does not need to use the key board to decode the key value, in which the complicated circuit and the I/O port are left out.

3 Description of Infrared Remote Theory

This chapter describes theory of infrared remote.

Infrared remote has many types which are decided by remote chip. In following description uPD6121G will be described.

Figure 3-1 describes the configuration of frame which is the remote code.

Figure 3-1: Remote Code

Figure 3-2 describes the encode “0” and “1”

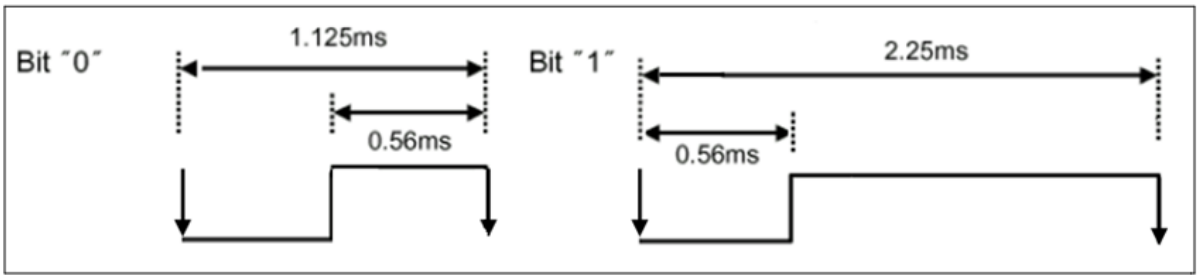

Figure 3-2: Encode Wave

For different remote the code and the configuration is different.

Normally the remote is generating high level except code status it will generate low and high level.

4 MB95F310 Infrared Remote Register

This chapter describes MB95F310 infrared remote register.

In MB95F310 series MCU remote function is realized by 8/16-BIT composite timer input capture module. Following Table 4-1 describes registers in composite timer input capture.

Table 4-1: All Register List

Register		Description
Timer0	T00CR0	Low 8bits of stats control register0
	T01CR0	High 8bits of stats control register0
Timer0	T00CR1	Low 8bits of stats control register1
	T01CR1	High 8bits of stats control register1
Timer0	T00DR	Low 8bits of data register
	T01DR	High 8bits of data register
Timer0	TMCR0	Timer mode control register
Timer1	T10CR0	Low 8bits of stats control register0
	T11CR0	High 8bits of stats control register0
Timer1	T10CR1	Low 8bits of stats control register1
	T11CR1	High 8bits of stats control register1
Timer1	T10DR	Low 8bits of data register
	T11DR	High 8bits of data register
Timer1	TMCR1	Timer mode control register

For detailed register please refer to MB95F310 Hardware Chapter 18.

Figure 4-1 describes these registers work condition in detail.

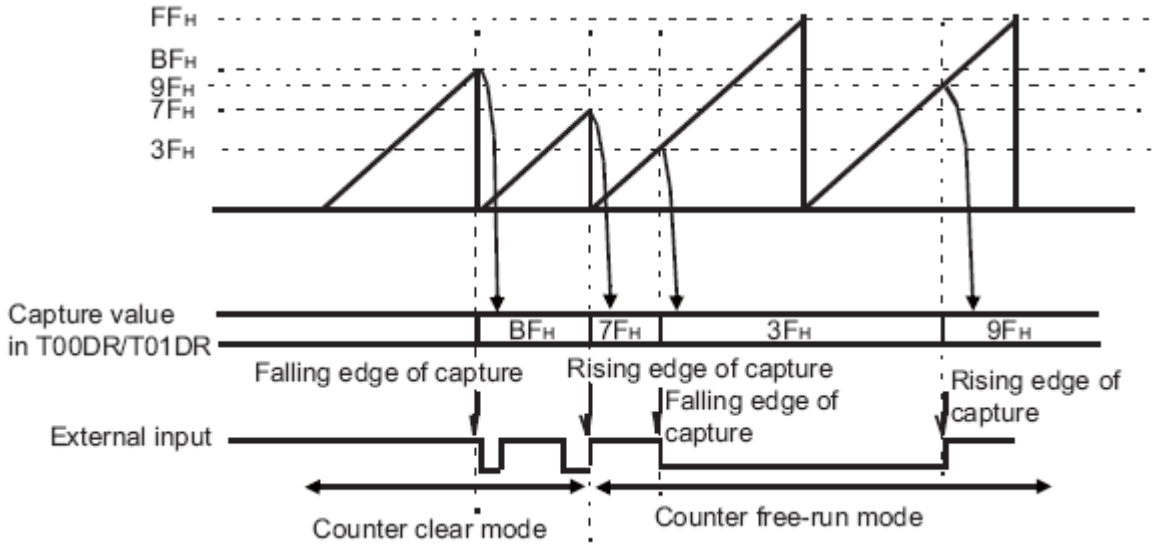

Figure 4-1: Register Work Condition

5 Infrared Remote Library Function List

This chapter introduces all functions in infrared remote library in project Simulate remote.prj which MCU are MB95F310.

Table 5-1 lists the infrared remote functions.

Table 5-1: Remote Capture Functions

Function name	Description
void initial_Remot(void)	Initializes remote capture register
__interrupt void inputcapture0 (void)	Interrupt to receive remote code
unsigned char Remote_detect(void)	According remote code to judge which key pressed

6 Infrared Remote Function Detail

This chapter introduces the detail of infrared remote functions.

6.1 Initial_Remot Function

Table 6-1 describes initial_Remot function.

Table 6-1: initial_Remot Function

Function name	**initial_I2C**
Function prototype	Void initial_Remot(void)
Behavior description	Initializes remote capture condition
Input parameter	None
Return value	None
Example	The library function sets clock use internal clock, counter interval is 0.64μs and falling edge trigger counter: initial_Remot();

If user wants to change capture condition, please refer to register T00CR0, T00CR1 and TMCR0.

6.2 Interrupt Function

Table 6-2 describes __interrupt function.

Table 6-2: Interrupt Function Description

Function prototype	__interrupt void inputcapture0 (void)
Behavior description	When captured a falling edge, the interrupt is occurred , and the pulse is received and the width of the pulse will be countered
Input parameter	None
Return value	None
Example	__interrupt void inputcapture0 ();

Different remote the configuration is different so the code is different; maybe it is 16-bit code maybe it is 32-bit code, user can review the code by oscilloscope. In this remote.prj the remote code is set to match 16-bit remote code.

6.3 Remote_detect Function

Table 6-3 describes Remote_detect function.

Table 6-3: Read_I2C Function

Function name	AD_Read
Function prototype	unsigned char Remote_detect(void)
Behavior description	Judges the code and then decides which key has been pressed according to the code
Input parameter	None
Return value	Key value of the remote
Example	[variable]= Remote_detect();

The key value can be decided by user's remote encode.

7 Usage Demo

This chapter describes something we must pay attention to when we use.

7.1 Hardware Design

For hardware design the following figure may be referred.

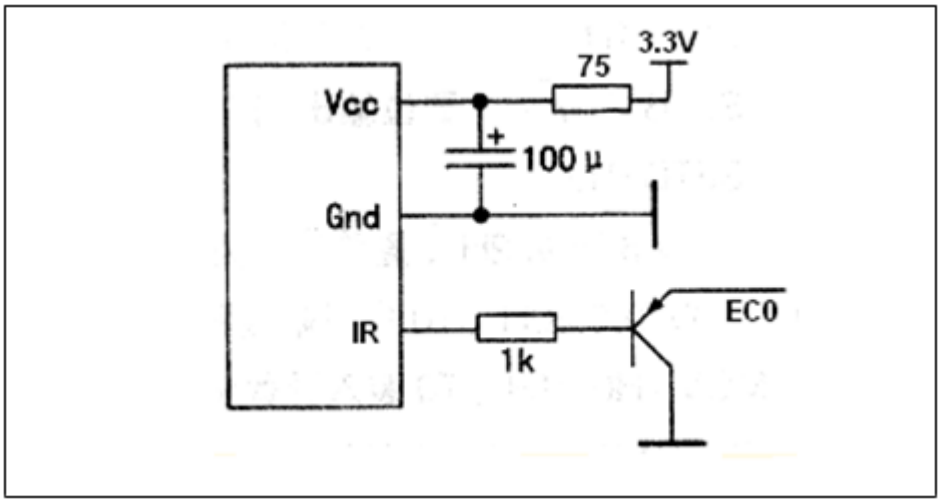

Figure 7-1: Remote Hardware Design

7.2 Steps for Software Design

- For infrared remote software design, initialization is the first step, which opens the interrupt, sets the interval time and trigger condition.

  Following setting is referred.

```
Void initial_Remot(void)
{
    TMCR0 = 0x00;
    T00CR0 = 0x4b;
    T00CR1 = 0xa0;
}
```

Figure 7-2: Initialization Design

- Detecting remote code is the important step. Each pulse is one bit of code. Each time enter into the interrupt one bit is detected. If the remote code is 16bits, after sixteen enter into interrupt, the key code is generated.
- The last step is key value judge. Different keys have different codes, so user can judge the key value by code which is generated in step two.

7.3 Steps for Adding Remote Library

When use this project please refer to following steps.

- First step is to add library to document, Figure 7-3 describes this step

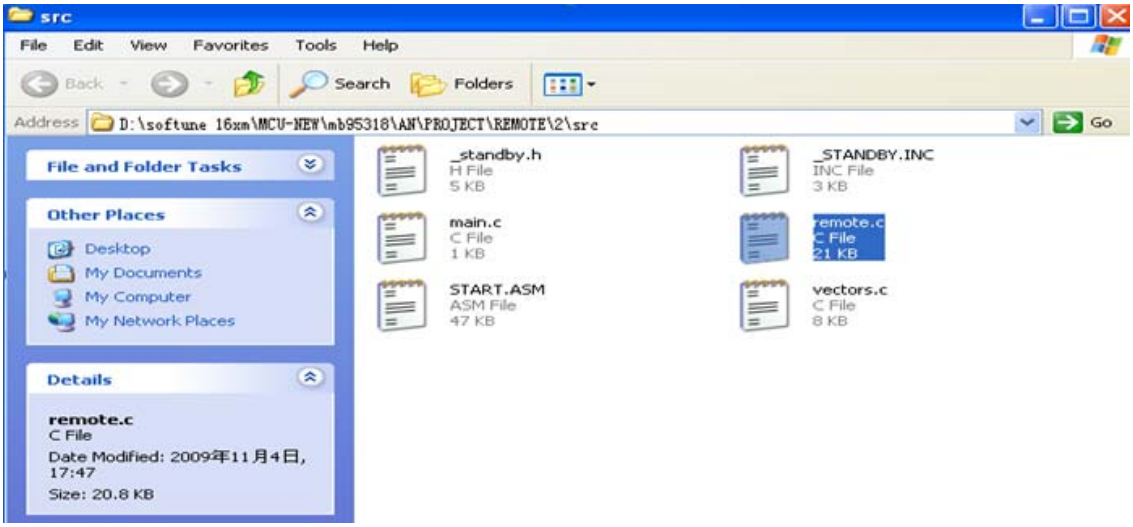

Figure 7-3: Library Use First Step

- Second step is to add function to project, Figure 7-4 describes this step

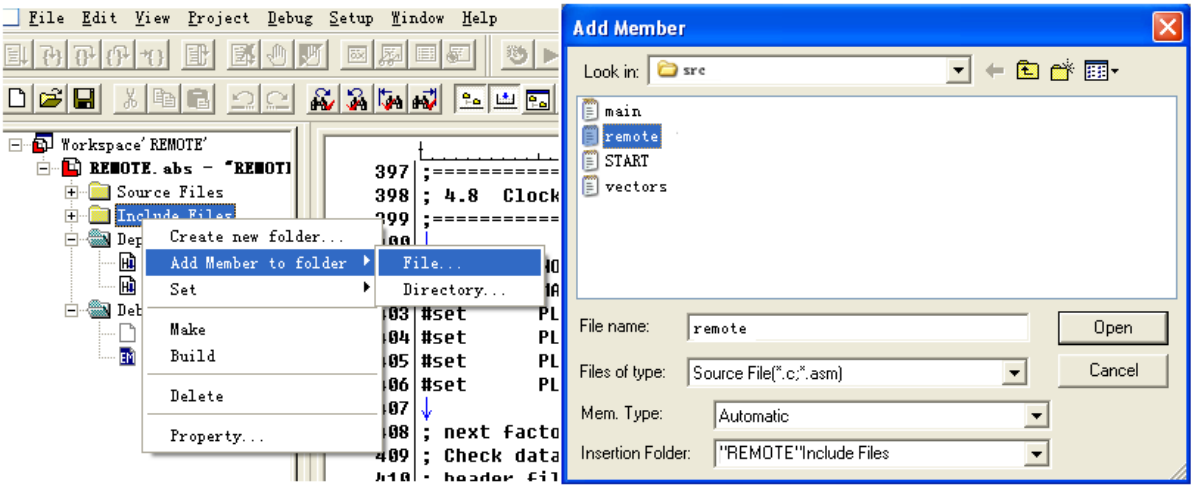

Figure 7-4: Library Use Second Step

- Third step is to add initial function to main.c, Figure 7-5 describes this step.

```
void main(void)
{
    unsigned char KeyVA;
    InitIrqLevels();
    __EI();

    Initial_Inter();

    while(1)
    {
        KeyVA = Remote_detect();
    }
}
```

Figure 7-5: Library Use Third Step

- Fourth step is to add interrupt function to vector.c, Figure 7 – 6 describes this step.

```
/*-----------------------------------------------------------------------
   Prototypes
   Add your own prototypes here. Each vector definition needs is proto-
   type. Either do it here or include a header file containing them.
-----------------------------------------------------------------------*/

__interrupt void DefaultIRQHandler (void);
__interrupt void inputcapture0 (void);
/*-----------------------------------------------------------------------
```

Figure 7 - 1 Library Use Fourth Step

- Fifth step is to add remote_detect function to main.c, Figure 7-7 describes this step.

```
 8 void main(void)
 9 {
10     unsigned char KeyVA;
11     InitIrqLevels();
12     __EI();
13
14     Initial_Inter();
15
16     while(1)
17     {
18         KeyVA = Remote_detect();
19     }
20 }
```

Figure 7-6: Library Use Fifth Step

- The sixth step is debugging, set a environment before debugging, Figure 7-8 describes this step

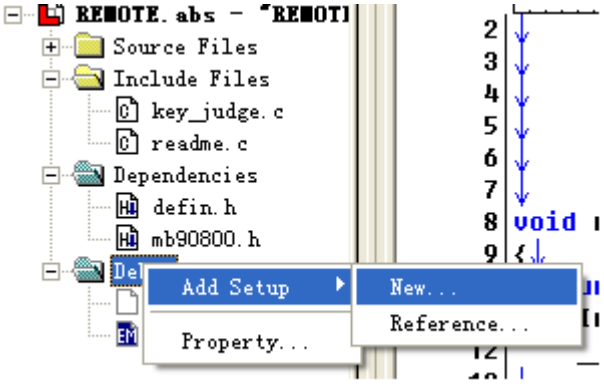

Figure 7-7: Library Use Sixth Step

For more condition please refer to Chapter 8.

8 Debug

This chapter describes how to debug the sample code on EV-board and what will happen when the code is running.

There is a simple project remote.prj for debugging. This project is based on our EV-board MB2146-450-E and the target MCU is MB95F310.
When debugging, the hardware linking please refer to Figure 7-1.
Press one key and the wave will be detected by oscilloscope as Figure 8-1.

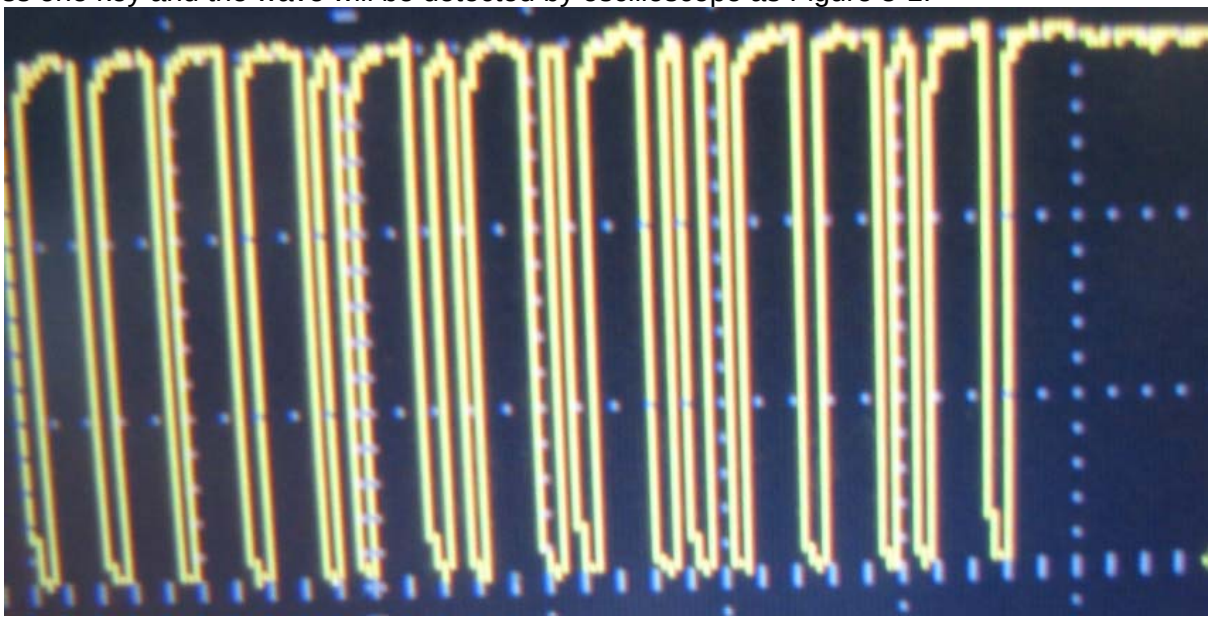

Figure 8-1: Key Code

The wide wave is binary “1”, and the short wave is binary “0”.

From this picture we can judge the remote code is “1111 0101 0100 1101” in binary

Run project, the key value will be read by Remote_detect function. And key value will be read out by global variable “key_Valu”. In this sample the **ENTER** key is pressed. Please refer to Figure 8-2 for detailed result.

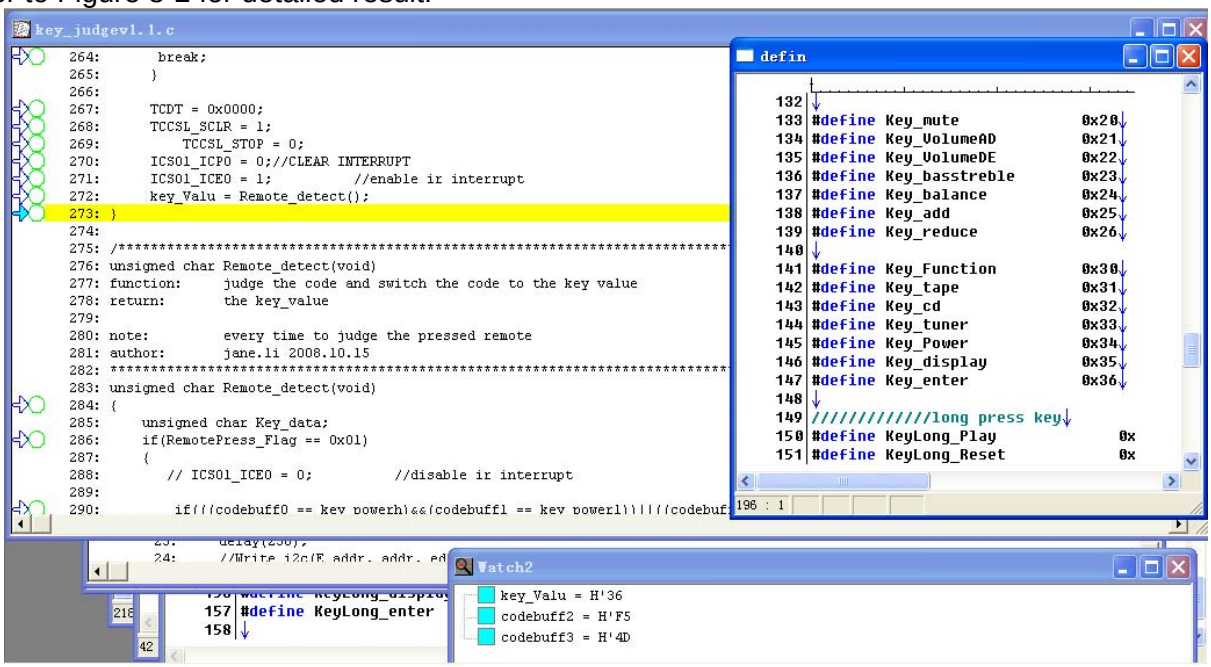

Figure 8-2: Debugging Description

9 Additional Information

For more information about how to use MB95310 EV-board, BGM Adaptor and SOFTUNE, please refer to EV-Board MB2146-450-E User Manual, or visit Websites:

English version:

http://www.fujitsu.com/cn/fsp/services/mcu/mb95/application_notes.html

Simplified Chinese Version:

http://www.fujitsu.com/cn/fss/services/mcu/mb95/application_notes.html

10 Appendix